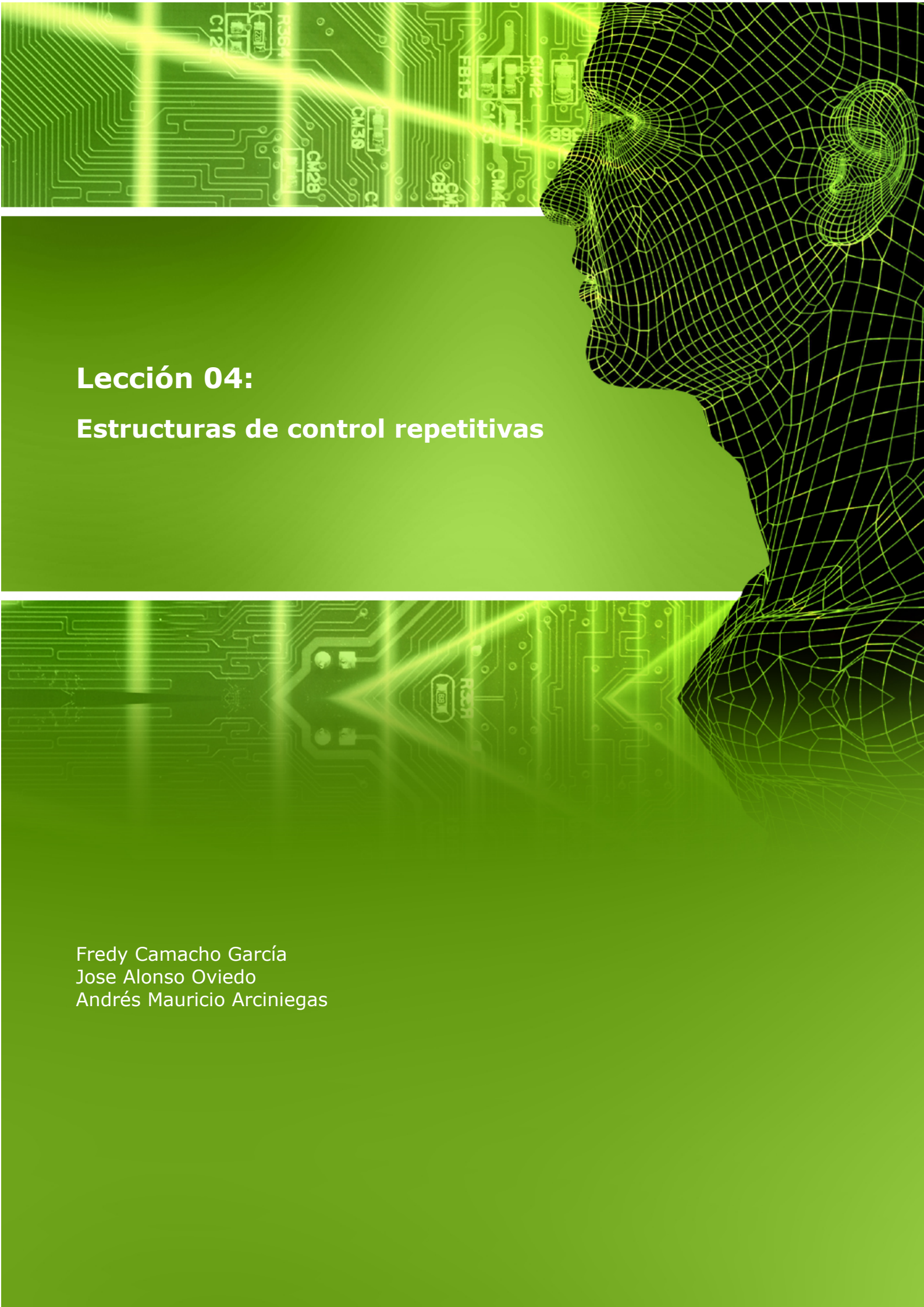


# Lección 04:

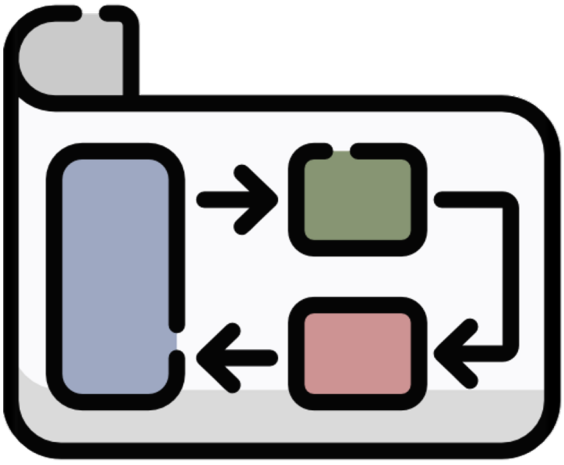
## Estructuras de control repetitivas

Fredy Camacho García
Jose Alonso Oviedo
Andrés Mauricio Arciniegas

# 4

## Estructuras de control repetitivas

Es muy común encontrar en la práctica algoritmos cuyas operaciones se deben ejecutar un número repetido de veces. Si bien las instrucciones son las mismas los datos sobre los cuales se opera si cambian. El conjunto de instrucciones que se ejecuta repetidamente se denomina **ciclo**.



Las estructuras de control repetitivas son aquellas que permiten ejecutar un conjunto de instrucciones varias veces, de acuerdo con una expresión lógica.

Mientras esta sea verdadera, el ciclo se ejecutará, sin embargo, este debe terminar de ejecutarse luego de un número finito de veces, por lo que en cada repetición es necesario evaluar alguna condición para decidir si se repite o no la ejecución del ciclo.

En principio, se van a analizar dos tipos de instrucciones repetitivas:

- while
- for

## Instrucción while

Es muy similar a la instrucción **if**, con la diferencia que **while** se devuelve y evalúa de nuevo la condición. Si continúa siendo verdadera, de nuevo ejecuta las instrucciones que están dentro de la estructura.

Recordemos la sintaxis de la instrucción if

```
if condición :
     instrucciones por ejecutar si la
     condición es verdadera
```

Las instrucciones que se encuentran **indentadas** en la instrucción **if** se ejecutan una sola vez.

La sintaxis de la instrucción **while** es similar:

```
while condición :
     instrucciones por ejecutar si la
     condición es verdadera
```

La diferencia es que una vez finaliza la ejecución de las instrucciones que se encuentran **indentadas** de la instrucción **while**, el flujo se devuelve al principio, es decir, a la condición y la evalúa de nuevo (verifica si aún es verdadera o no). Si es verdadera, ejecuta de las instrucciones y repite el procedimiento. Si es falso, termina la ejecución y continua con las instrucciones que están después de la misma.

A continuación, se presentarán algunos ejemplos en Python haciendo uso de esta instrucción.

**Ejemplo 1**.

Implementar un algoritmo que muestre los primeros 10 números naturales

**Solución:**

- *Datos Entrada*: *Ninguno*
- *Datos Salida*: numero

```
# Instrucciones Repetitivas
# Ejemplo 1

numero = 1

while numero <= 10 :
  print( numero )
  numero = numero + 1

print( "Fin del algoritmo" )
```

Observe que:

- La variable *numero* inicia en 1
- En la instrucción **while**, se evalúa el valor de la variable número. Si esta es menor a 10, se ejecuta las instrucciones que se encuentran. A estas instrucciones se les llama **ciclo**. Aquí se muestra el número y después se incrementa en 1 su valor.
- Cuando termina de ejecutarse las instrucciones que están indentadas, la ejecución regresa al principio, y de nuevo evalúa la condición. Si esta es verdad de nuevo, vuelve y se ejecuta el ciclo.
- Esto se repite hasta que la condición sea falsa.
- Para que la condición se vuelva falsa en algún momento, el dato que se evalúa en la condición debe modificarse dentro del ciclo. Esto garantiza que no se vuelva infinito el ciclo.

**Ejemplo 2**.

Leer 5 números utilizando una instrucción repetitiva y mostrar el número leído.

**Solución**

- *Datos Entrada*: numero
- *Datos Salida*: numero

```
# Ejemplo 2 - Leer 5 números utilizando una
instrucción repetitiva y mostrarlo posteriormente

cantidad = 1

while cantidad <= 5 :
  numero = int( input() )
  print( numero )

  cantidad = cantidad + 1

print( "Fin del algoritmo" )

# cantidad está haciendo una función de contador
# (cuenta de 1 en 1, inicia en 1 y llega hasta 5)
```

**Ejemplo 3.**

Leer 10 números, calcular y mostrar su cuadrado. Utilice una instrucción repetitiva para ello.

**Solución**

- *Datos Entrada*: numero
- *Datos Salida*: cuadrado

El ciclo consiste en:

Leer un número
Calcular el cuadrado del número leído
Mostrar el cuadrado obtenido
Incrementar en 1 la variable que se encarga de contar (para que cuente y finalice en algún momento)

Si no existieran las instrucciones repetitivas:

```
# Ejemplo 3 - Sin instrucciones repetitivas

print( "Ingrese un número: " )
numero = int( input() )
cuadrado = numero ** 2
print( "El cuadrado de ", numero, " es ", cuadrado )

print( "Ingrese un número: " )
numero = int( input() )
cuadrado = numero ** 2
print( "El cuadrado de ", numero, " es ", cuadrado )

print( "Ingrese un número: " )
numero = int( input() )
cuadrado = numero ** 2
print( "El cuadrado de ", numero, " es ", cuadrado )

# ... Se repite 6 veces más

print( "Ingrese un número: " )
numero = int( input() )
cuadrado = numero ** 2
print( "El cuadrado de ", numero, " es ", cuadrado )

print( "Fin del algoritmo" )
```

Con instrucciones repetitivas:

```
# Ejemplo 3 con instrucciones repetitivas
contador = 1

while contador <= 10 :
  print( "Ingrese un número: " )
  numero = int( input() )
  cuadrado = numero ** 2
  print( "El cuadrado de ", numero, " es ", cuadrado )

  contador = contador + 1

print( "Fin del algoritmo" )
```

**Ejemplo 4.**

Leer el sueldo actual de 4 trabajadores, calcular y mostrar su nuevo sueldo. El nuevo sueldo equivale al sueldo actual incrementado en un 10%.

**Solución**

- *Datos Entrada*: sueldo1, sueldo2, sueldo3, sueldo4
- *Datos de Salida*: $nuevoSueldo_1$, $nuevoSueldo_2$, $nuevoSueldo_3$, $nuevoSueldo_4$

El ciclo consiste en:

Leer el sueldo actual de un trabajador
Calcular el nuevo sueldo del trabajador
Mostrar el nuevo sueldo que se ha calculado
Incrementar en 1 la variable que se encarga de contar (para que cuente y finalice en algún momento)

```
# Ejemplo 4. Leer el sueldo actual de 4 trabajadores,
calcular y
# mostrar su nuevo sueldo. El nuevo sueldo equivale al
sueldo actual
# incrementado en un 10%.

# Datos Entrada:  sueldo1, sueldo2, sueldo3, sueldo4
# Datos de Salida:  nuevoSueldo1, nuevoSueldo2,
nuevoSueldo3, nuevoSueldo4

contador = 1

while contador <= 4 :
  print( "Ingrese el sueldo trabajador ", contador, " : ")
  sueldo = float( input( ) )
  nuevoSueldo = sueldo * 1.10
  print( "El nuevo sueldo es: ", nuevoSueldo)

  contador = contador + 1

print( "Fin del algoritmo" )
```

**Ejemplo 5.**

Escribir un algoritmo que permita leer la nota final de 8 estudiantes de un curso y determinar si aprobó o reprobó el mismo. Las notas leídas están entre 0.0 y 5.0. El curso se aprueba con una nota mínima de 3.0

**Solución**

- *Datos Entrada*: $nota_1$, $nota_2$, …, $nota_8$
- *Datos de Salida*: $mensaje_1$, $mensaje_2$, …, $mensaje_8$

El ciclo consiste en:

Leer la nota final de un estudiante
Determinar si aprobó o reprobó el curso (instrucción condicional doble)

Incrementar en 1 la variable que se encarga de contar (para que cuente y finalice en algún momento)

```
# Ejemplo 5
contador = 1

while contador <= 8 :
  print( "Ingrese la nota final del curso: " )
  nota = float( input() )

  if nota >= 3.0 :
    print( "Aprobó el curso" )
  else:
    print( "Reprobó el curso" )

  contador = contador + 1

print( "Fin del algoritmo" )
```

**Ejemplo 6**.

Escribir un script que lea el sueldo de un grupo de 5 empleados de una empresa, calcule y muestre su nuevo sueldo de acuerdo con la siguiente política:

- Si el sueldo es inferior a $1.000.000, entonces el aumento es del 5%
- Si el sueldo es igual o superior a $1.000.000, entonces el aumento es del 2%

**Solución**

- *Datos Entrada*: $sueldo_1$, $sueldo_2$, …, $sueldo_5$
- *Datos de Salida*: $nuevoSueldo_1$, $nuevoSueldo_2$,…, $nuevoSueldo_5$

Para un empleado:
Leer el sueldo actual del empleado
Calcular el nuevo sueldo de acuerdo con la política establecida
Mostrar el nuevo sueldo del empleado
(Repetir 5 veces)

```
# Ejemplo 6

contador = 1

while contador <= 5 :
  print( "Ingrese el sueldo del empleado ", contador, ":" )
  sueldo = float( input( ) )

  if sueldo < 1000000 :
    nuevoSueldo = sueldo + (sueldo * 0.05)
  else:
    nuevoSueldo = sueldo + (sueldo * 0.02)

  print( "El nuevo sueldo es: ", nuevoSueldo )

  contador = contador + 1

print( "Fin de Algoritmo" )
```

**Ejemplo 7.**

Modifique el ejercicio anterior para que se pida al usuario el número de empleados a procesar

**Solución**

- *Datos Entrada*: cantidadEmpleados, $sueldo_1$, $sueldo_2$, …, $sueldo_5$
- *Datos de Salida*: $nuevoSueldo_1$, $nuevoSueldo_2$,…, $nuevoSueldo_5$

```
# Ejemplo 7
contador = 1

print( "Ingrese la cantidad de empleados: " )
cantidadEmpleados = int( input( ) )

while contador <= cantidadEmpleados :
  print( "Ingrese el sueldo del empleado ", contador, ":" )
  sueldo = float( input( ) )

  if sueldo < 1000000 :
    nuevoSueldo = sueldo + (sueldo * 0.05)
  else:
    nuevoSueldo = sueldo + (sueldo * 0.02)

  print( "El nuevo sueldo es: ", nuevoSueldo )

  contador = contador + 1

print( "Fin de Algoritmo" )
```

**Ejemplo 8.**

Escribir un programa que lea la calificación final de N estudiantes (N es un valor ingresado por el usuario) y determine cuántos aprobaron. La calificación es un valor entre 0.0 y 5.0, y para aprobar un curso la nota mínima es 3.0.

Por ejemplo: si N = 6

| | |
|---|---|
| | Aprobados = 0 (Inicialmente) |
| nota1 = 3.5 | aprobados = 1 |
| nota2 = 4.2 | aprobados = 2 |
| nota3 = 2.5 | aprobados = 2 |
| nota4 = 1.0 | aprobados = 2 |
| nota5 = 4.7 | aprobados = 3 |
| nota6 = 2.9 | aprobados = 3 |

Aprobaron 3 estudiantes

**Solución**

- *Datos Entrada*: n, $nota_1$, $nota_2$, …, $nota_n$
- *Datos de Salida*: aprobados

```
# Ejemplo 8

print( "Ingrese el número de estudiantes: " )
n = int( input( ) )

contador = 1
aprobados = 0

while contador <= n :
  print( "Ingrese la nota del estudiante ", contador, ":")
  nota = float( input( ) )

  if nota >= 3.0 :
    aprobados = aprobados + 1

  contador = contador + 1

print( "La cantidad de alumnos que aprobaron es: ",
aprobados )

print( "Fin del algoritmo" )
```

**Ejemplo 9.**

Modifique el ejemplo 8 para que cuente el número de estudiantes que aprobaron, y, también el número de estudiantes que reprobaron el curso.

**Solución**

- *Datos Entrada*: n, $nota_1$, $nota_2$, …, $nota_n$
- *Datos de Salida*: aprobados, reprobados

```
# Ejemplo 9

print( "Ingrese el número de estudiantes: " )
n = int( input( ) )

contador = 1
aprobados = 0
reprobados = 0

while contador <= n :
  print( "Ingrese la nota del estudiante ", contador, ":")
  nota = float( input( ) )

  if nota >= 3.0 :
    aprobados = aprobados + 1
  else:
    reprobados = reprobados + 1

  contador = contador + 1

print( "La cantidad de alumnos que aprobaron es: ",
aprobados )
print( "La cantidad de alumnos que reprobaron es: ",
reprobados )

print( "Fin del algoritmo" )
```

**Ejemplo 10.**

Escribir un script que permita leer N números y calcular su suma.

**Solución**

Considere los siguientes datos de ejemplo:

N = 5

| contador | numero | suma |
|---|---|---|
| 1 | 8 | 0 + 8 = 8 |
| 2 | 12 | 8 + 12 = 20 |
| 3 | 15 | 20 + 15 = 35 |
| 4 | 7 | 35 + 7 = 42 |
| 5 | 20 | 42 + 20 = 62 |

Se puede apreciar que,

Se lee un número
Se acumula
Se incrementa el contador en 1

En términos de instrucciones:

```
numero = int( input() )
suma = suma + numero
contador  = contador + 1
```

- *Datos Entrada*: n, $numero_1$, $numero_2$,…, $numero_N$
- *Datos de Salida*: suma

```
# Ejemplo 10
print( "Ingrese la cantidad de números:" )
n = int( input() )

contador = 1
suma = 0

while contador <= n :
  print( "Ingrese el número: " )
  numero = int( input() )

  # acumulamos el número leído
  suma = suma + numero
  contador = contador + 1

print( "La suma es: ", suma )
print( "Fin del algoritmo" )
```

**Ejemplo 11.**

En una empresa se requiere de un programa que permita leer el sueldo de N trabajadores y calcular el total de la nómina (la suma de los sueldos).

**Solución**

- *Datos Entrada*: n, $sueldo_1$, $sueldo_2$,…, $sueldo_n$
- *Datos Salida*: suma

```
# Ejemplo 11

print( "Ingrese la cantidad de empleados:" )
n = int( input() )

contador = 1
suma = 0

while contador <= n :
  print( "Ingrese el sueldo del empleado", contador, ":" )
  sueldo = int( input() )

  # acumulamos el sueldo leído
  suma = suma + sueldo

  contador = contador + 1

print( "El total de la nómina es: ", suma )
print( "Fin del algoritmo" )
```

**Ejemplo 12.**

Modifique el ejemplo anterior para que calcule el promedio de sueldo por empleado. El promedio se calcula como la división de la suma de un conjunto de valores entre la cantidad.

**Solución**

- *Datos Entrada*: n, $sueldo_1$, $sueldo_2$,…, $sueldo_n$
- *Datos Salida*: suma, promedio

```
# Ejemplo 12

print( "Ingrese la cantidad de empleados:" )
n = int( input() )

contador = 1
suma = 0

while contador <= n :
  print( "Ingrese el sueldo del empleado", contador, ":" )
  sueldo = int( input() )

  # acumulamos el sueldo leído
  suma = suma + sueldo

  contador = contador + 1

# Cálculo del promedio
promedio = suma / n

print( "El total de la nómina es: ", suma )
print( "El promedio de sueldos de la empresa es: ",
promedio )

print( "Fin del algoritmo" )
```

**Ejemplo 13.**

Leer N números y determinar cuál es el mayor de estos.

**Solución**

- *Datos Entrada*: n, $numero_1$, $numero_2$,…, $numero_n$
- *Datos Salida*: mayor

*Análisis del caso*

Suponga los siguientes datos de ejemplo

N = 5

| contador | numero | mayor | |
|---|---|---|---|
| | | 0 | Mínimo valor posible |
| 1 | 10 | 10 | |
| 2 | 15 | 15 | |
| 3 | 7 | 15 | |
| 4 | 20 | 20 | |
| 5 | 14 | 20 | |

Observe que a medida que se lee un número se va comparando con el que hasta el momento es el mayor de los leídos previamente. Una vez se han leído todos los datos, se tiene el mayor de estos.

En Python la implementación es la siguiente:

```
# Ejemplo 13

print( "Ingrese la cantidad de números:" )
n = int( input() )
```

```
contador = 1
mayor = 0

while contador <= n :
  print( "Ingrese un número: " )
  numero = int( input() )

  if numero > mayor :
    mayor = numero

  contador = contador + 1

print( "El mayor es: ", mayor )

print( "Fin del algoritmo" )
```

## Ejercicios Propuestos

### Ejercicio 1

¿Cómo solucionar la siguiente situación?

```
Ingrese la cantidad de números:
5
Ingrese un número:
-6
Ingrese un número:
-2
Ingrese un número:
-10
Ingrese un número:
-7
Ingrese un número:
-8
El mayor es:  0
Fin del algoritmo
```

+ Código + Texto

### Ejercicio 2

Leer N números y determinar cuál es el menor de estos.

**Ejemplo 14**

Escribir un algoritmo que permita leer la nota final de N estudiantes de un curso, y determinar cuántos aprobaron el curso, cuantos reprobaron el curso, la mayor calificación obtenida, la menor calificación obtenida y el promedio de notas del curso. Las calificaciones son valores entre 0.0 y 5.0; y para aprobar el curso, la nota mínima debe ser 3.0.

**Solución**

- *Datos Entrada*: n, $nota_1$, $nota_2$, $nota_3$, …, $nota_n$
- *Datos de Salida*: aprobados, reprobados, mayor, menor, promedio

```
# Ejercicio 14

print( "Ingrese la cantidad de estudiantes: " )
n = int( input() )

contador = 1
aprobados = 0
reprobados = 0
mayor = 0.0
menor = 5.0
suma = 0.0

while contador <= n :
  print( "Ingrese nota del estudiante", contador )
  nota = float( input() )

  if nota >= 3.0 :
    aprobados = aprobados + 1
  else :
    reprobados = reprobados + 1
```

```
    if nota > mayor :
      mayor = nota

    if nota < menor :
      menor = nota

    suma = suma + nota

    contador = contador + 1

promedio = suma / n

print( "La cantidad de aprobados es: ", aprobados )
print( "La cantidad reprobados es: ", reprobados )
print( "La mayor nota es: ", mayor )
print( "La menor nota es: ", menor )
print( "El promedio de notas es: ", promedio )

print( "Fin del algoritmo" )
```

## Instrucciones repetitivas controladas por centinela

Un centinela se emplea cuando en un algoritmo o programa requiere repetir la ejecución de un grupo de instrucciones, pero, no se conoce la cantidad de veces que hay que hacerlo. En este caso, se debe conocer cuál es el caso que permite la finalización y establecer la expresión lógica correspondiente.

**Ejemplo 15.**

Leer una serie de gastos que ha tenido una persona durante un mes, calcular y mostrar la suma de estos. La finalización de la captura de gastos se hará con un cero (0).

Si se hiciera conociendo de antemano el número de gastos, la solución sería

```
# Ejemplo 15
print( "Ingrese el número de gastos: " )
n = int( input() )

contador = 1
suma = 0.0

while contador <= n :
  print( "Ingrese un gasto: " )
  gasto = float( input() )

  suma = suma + gasto
  contador = contador + 1

print( "El total de gastos es: ", suma )
print( "Fin del algoritmo" )
```

En este script se implementó una **instrucción repetitiva controlada por contador** (para que repita el ciclo el número de veces que se especificó).

Para dar respuesta al requisito establecido, se plantea la siguiente solución: Cuando no se conoce el número de repeticiones, se debe conocer cuando finaliza la repetición. En el ejemplo planteado, esto ocurre cuando ingresa un cero (0) como valor de gasto.

Así entonces el código se modifica de la siguiente forma:

```
# Ejemplo 15
suma = 0.0

# Se lee el primer gasto
print( "Ingrese el gasto : " )
gasto = float( input() )
```

```
while gasto != 0 :          # El control se hace a
partir del valor de finalización (centinela)
  suma = suma + gasto

  print( "Ingrese el gasto : " )
  gasto = float( input() )

print( "El total de gastos es: ", suma )
print( "Fin del algoritmo" )
```

Observe que la instrucción **while** se plantea de la siguiente forma: `while gasto != 0 :`

Esto significa que la instrucción repetitiva va a iterar la ejecución del ciclo mientras el valor de la variable gasto sea distinto de cero. Cuando se vuelva igual a cero, la repetición finalizará, sin embargo, esto permite que repita cualquier cantidad de veces la ejecución del ciclo. Esta variable se conoce como **centinela**, por tanto, se habla de una **instrucción repetitiva controlada por centinela.**

**Ejemplo 16.**

Supóngase que en una reciente elección hubo cuatro candidatos (con identificadores 1, 2, 3, 4). Usted habrá de encontrar, mediante un programa, el número de votos correspondiente a cada candidato, el porcentaje que obtuvo respecto al total de los votantes y el candidato ganador de la elección. El usuario ingresará por teclado los votos de manera desorganizada, tal y como se obtuvieron en la elección, el final de datos está representado por un cero. Observe, como ejemplo, la siguiente lista:

1 3 1 4 2 2 1 4 1 1 1 2 1 3 1 4 0

Donde 1 representa un voto para el candidato 1; 3 un voto para el candidato 3; y así sucesivamente.

**Solución:**

**Datos Entrada**: $voto_1$, $voto_2$, $voto_3$, $voto_4$, $voto_5$, …, 0
**Datos de Salida**: candidato1, candidato2, candidato3, candidato4, porcentajeCan1, porcentajeCan2, porcentajeCan3, porcentajeCan4, ganador

Un porcentaje se obtiene dividiendo una cantidad sobre un total, y multiplicando por 100.

Por ejemplo, si el candidato 1 obtuvo 20 votos y en total fueron 80 votos, entonces, el porcentaje de votación del candidato 1 es: 20/80 * 100 = 0,25 * 100 = 25%

```
candidato1 = 0          # Cantidad de votos del candidato 1
candidato2 = 0          # Cantidad de votos del candidato 2
candidato3 = 0          # Cantidad de votos del candidato 4
candidato4 = 0          # Cantidad de votos del candidato 4

cantidad = 0            # Cantidad total de votos

porcentajeCan1 = 0.0    # Porcentaje obtenido por candidato 1
porcentajeCan2 = 0.0    # Porcentaje obtenido por candidato 2
porcentajeCan3 = 0.0    # Porcentaje obtenido por candidato 3
porcentajeCan4 = 0.0    # Porcentaje obtenido por candidato 4

# Se lee el primer voto
voto = int( input( "Ingrese el voto: ") )

# Mientras el voto no sea cero(0) determine a que candidato
# le pertenece
while voto != 0 :
  # Se cuenta el voto como uno más
  cantidad = cantidad + 1
```

```
    # Se determina a que candidato se le suma el voto
    if voto == 1 :
      candidato1 = candidato1 + 1
    elif voto == 2 :
      candidato2 = candidato2 + 1
    elif voto == 3 :
      candidato3 = candidato3 + 1
    else:
      candidato4 = candidato4 + 1

    # Se lee el siguiente voto
    voto = int( input( "Ingrese el voto: ") )

# Fuera de la instrucción repetitiva ...
# Se calculan los porcentajes de votación por candidato
porcentajeCan1 = candidato1 / cantidad * 100
porcentajeCan2 = candidato2 / cantidad * 100
porcentajeCan3 = candidato3 / cantidad * 100
porcentajeCan4 = candidato4 / cantidad * 100

# El cálculo del ganador debe hacerse aquí: propuesto

# Resultados
print( "El candidato 1 tuvo ", candidato1, " votos y
representa el ", porcentajeCan1, "%")
print( "El candidato 2 tuvo ", candidato2, " votos y
representa el ", porcentajeCan2, "%")
print( "El candidato 3 tuvo ", candidato3, " votos y
representa el ", porcentajeCan3, "%")
print( "El candidato 4 tuvo ", candidato4, " votos y
representa el ", porcentajeCan4, "%")
print( "El ganador es: " )        # Pendiente
```

Una ejecución de prueba puede ser como la siguiente:

```
Ingrese el voto: 1
Voto para el candidato 1
Ingrese el voto: 3
Ingrese el voto: 1
Voto para el candidato 1
Ingrese el voto: 4
Ingrese el voto: 2
Ingrese el voto: 2
Ingrese el voto: 1
Voto para el candidato 1
Ingrese el voto: 4
Ingrese el voto: 1
Voto para el candidato 1
Ingrese el voto: 1
Voto para el candidato 1
Ingrese el voto: 1
Voto para el candidato 1
Ingrese el voto: 2
Ingrese el voto: 1
Voto para el candidato 1
Ingrese el voto: 3
Ingrese el voto: 1
Voto para el candidato 1
Ingrese el voto: 4
Ingrese el voto: 0
El candidato 1 tuvo  8  votos y representa el  50.0 %
El candidato 2 tuvo  3  votos y representa el  18.75 %
El candidato 3 tuvo  2  votos y representa el  12.5 %
El candidato 4 tuvo  3  votos y representa el  18.75 %
El ganador es:
```

## Instrucción repetitiva For

Permite establecer instrucciones de repetición controladas por **contador**. No aplica para control por centinela.

La intención de la instrucción **for** es separar el ciclo del manejo del contador de iteraciones. Lo anterior significa que mientras que con la instrucción **while**, se debe incluir en el ciclo la siguiente instrucción:

```
contador = contador + 1
```

Con la instrucción **for** no es necesario dado que la misma estructura se encarga de hacerlo.

### Sintaxis de la instrucción for

```
for contador in range( inicio, fin, avance )
     ciclo
```

El valor de contador inicia en el valor especificado en `inicio` y termina en `fin - 1`.

### Ejemplo 1.

```
for contador in range( 3, 10, 1 ) :
  print( contador )
```

En pantalla se observa:

```
3
4
5
6
7
8
9
```

### Ejemplo 2.

```
for contador in range( 3, 10, 2 ) :
  print( contador )
```

En pantalla se observa:

```
3
5
7
9
```

**Ejemplo 3**

```
for contador in range( 3, 10, 3 ) :
  print( contador )
```

En pantalla se observa:

```
3
6
9
```

**Ejemplo 4**

```
for contador in range( 3, 10, 4 ) :
  print( contador )
```

En pantalla se observa:

```
3
7
```

El avance (incremento del contador) por defecto es 1. Esto quiere decir que la instrucción for se puede definir de la siguiente forma:

```
for contador in range( inicio, fin )
     ciclo
```

Al no especificar el avance, ese se hace de 1 en 1.

**Ejemplo 5**

```
# Instrucción for

for contador in range( 1, 5 ):
  print( contador )
```

El resultado despliega el valor de contador en pantalla desde 1 hasta (5-1), es decir, hasta 4.

```
for contador in range( 1, 5 ) :
  print( contador )

1
2
3
4
```

**Ejemplo 6**

```
for contador in range( 3, 10 ) :
  # Si no especifica el tercer parámetro de range()
  # se incrementa contador en 1
  print( contador )
```

En pantalla se observa:

```
3
4
5
6
7
8
9
```

**Ejemplo 7**

```
for contador in range( 1, 6 ) :
  print( contador )
```

En pantalla se observa:

```
1
2
3
4
5
```

Existe una tercera forma de expresar el rango en que el contador tomará valores:

```
for contador in range( fin )
     ciclo
```

Al no especificar el inicio, el contador comienza en 0, mientras que el avance se hace de 1 en 1. El contador tomará valores hasta `fin - 1`.

**Ejemplo 8**.

```
for contador in range( 6 ) :
  print( contador )
```

En pantalla se observa:

```
0
1
2
3
4
5
```

**Ejemplo 9.**

Escribir un algoritmo que permita leer la nota final de N estudiantes de un curso, y determinar cuántos aprobaron el curso, cuantos reprobaron el curso, la mayor calificación obtenida, la menor calificación obtenida y el promedio de notas del curso. Las calificaciones son valores entre 0.0 y 5.0; y para aprobar el curso, la nota mínima debe ser 3.0.
Nota. *Debe emplear la instrucción for*

**Solución**

- *Datos Entrada*: n, $nota_1$, $nota_2$, $nota_3$, …, $nota_n$
- *Datos Salida*: aprobados, reprobados, mayor, menor, promedio

```
# Ejemplo 9 - Instrucción for

print( "Ingrese cantidad de estudiantes:  " )
n = int( input() )

aprobados = 0
reprobados = 0
mayor = 0.0
menor = 5.0
suma = 0.0

for contador in range( n ) :
  print( "Ingrese nota estudiante", (contador + 1))
  nota = float( input() )

  if nota >= 3.0 :
    aprobados = aprobados + 1
  else :
    reprobados = reprobados + 1
```

```
    if nota > mayor :
      mayor = nota

    if nota < menor :
      menor = nota

    suma = suma + nota


promedio = suma / n

print( "Cantidad de aprobados: ", aprobados )
print( "Cantidad de reprobados: ", reprobados )
print( "La mayor nota es: ", mayor )
print( "La menor nota es: ", menor )
print( "El promedio de notas es: ", promedio )

print( "Fin del algoritmo" )
```

En pantalla se observa lo siguiente:

```
Ingrese la cantidad de estudiantes del curso:
5
Ingrese nota estudiante 1
3.5
Ingrese nota estudiante 2
4.2
Ingrese nota estudiante 3
2.7
Ingrese nota estudiante 4
3.1
Ingrese nota estudiante 5
4.5
Cantidad de aprobados:  4
Cantidad de reprobados:  1
La mayor nota es:  4.5
La menor nota es:  2.7
El promedio de notas es:  3.6
Fin del algoritmo
```

Otra forma de establecer la instrucción **for** es la siguiente (observe la instrucción **print** a continuación). Los cambios se encuentran resaltados en amarillo.

```
# Ejemplo 9 - Instrucción for
print( "Ingrese la cantidad de estudiantes del curso:  " )
n = int( input() )

aprobados = 0
reprobados = 0
mayor = 0.0
menor = 5.0
suma = 0.0

for contador in range( 1, n+1 ) :
  print( "Ingrese la nota final del estudiante", contador )
  nota = float( input() )

  if nota >= 3.0 :
    aprobados = aprobados + 1
  else :
    reprobados = reprobados + 1

  if nota > mayor :
    mayor = nota

  if nota < menor :
    menor = nota

  suma = suma + nota

promedio = suma / n

print( "Cantidad de aprobados: ", aprobados )
print( "Cantidad de reprobados: ", reprobados )
print( "La mayor nota es: ", mayor )
print( "La menor nota es: ", menor )
print( "El promedio de notas es: ", promedio )
print( "Fin del algoritmo" )
```

## Ejercicios Propuestos

### Ejercicio 1

Modifique el ejercicio anterior para que se indique cuál es el estudiante con la mayor nota y cuál es el estudiante con la menor nota.

### Ejercicio 2

Escribir un algoritmo (script) que permita generar la siguiente sucesión:

10, 9, 8, 7, 6, …, 1

### Ejercicio 3

Escribir un algoritmo (script) que calcule la suma de la siguiente sucesión (serie)

1 + 2 + 3 + 4 + … + N

### Ejercicio 4

Escribir un algoritmo (script) que calcule la suma de la siguiente sucesión (serie)

$$1 - 2 + 3 - 4 + 5 - 6 + \cdots \pm N$$

### Ejercicio 5

Escribir un algoritmo (script) que calcule la suma de la siguiente sucesión (serie)

$$1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \frac{1}{5} - \frac{1}{6} + \cdots \pm \frac{1}{N}$$

**Ejercicio 6**

Escribir un algoritmo (script) que genere la siguiente sucesión:

1, 3, 6, 8, 11, 13, 16, 18, …

**Ejercicio 7**

Escribir un algoritmo (script) que permita generar los primeros N términos de la serie de Fibonacci. La serie se caracteriza porque cada término es la suma de los dos anteriores. Los dos primeros términos son 0 y 1. La serie es:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144